ICE 8 Brian Bauman

1. Using the DiGraph API, write a function to compute the average inDegree of a given DiGraph

double avgInDegree(DiGraph g) {

double total = 0.0;

DiGraph reversedGraph = g.reverse();

for (int v : g.V()) {

int inDegreeCount = 0;

for (int w : g.adj(v))

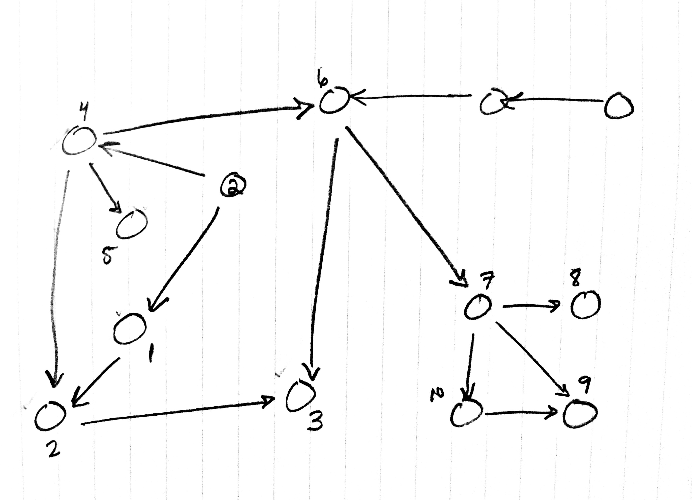
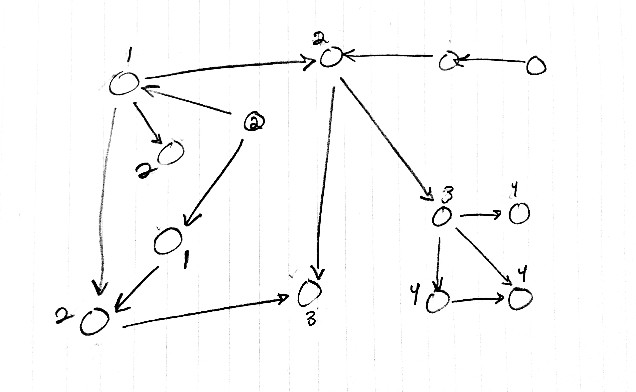
inDegreeCount++;

total += inDegreeCount;

}

return total / g.V().length;

}

1. Do a DFS of the graph shown starting at vertex 2. Label the verticies in the order in which they are visited.
2. Do a BFS of the graph shown starting at vertex 2. Label the vertices by their depth from the root (2).
3. Paraphrase any two non-consecutive questions asked during class (by anyone) along with the answer.

1) At 00:27:52, you ask the class what the other two representations of graphs are. The answer was edge list and adjacency matrix.

2) At 01:48:48, you ask what the API would look like to solve the problem of determining whether or not a graph has a cycle. The answer is that you’ll follow the design pattern set by the author of the book, creating a new class that performs a search through the graph to determine the answer.